

# Minimizing Total Busy Time with Application to Energy-efficient Scheduling of Virtual Machines in IaaS clouds

Nguyen Quang-Hung, Nam Thoai

*Faculty of Computer Science and Engineering,  
HCMC University of Technology, VNU-HCM  
Ho Chi Minh City, Vietnam  
Email: {hungnq2,nam}@cse.hcmut.edu.vn*

**Abstract**—Infrastructure-as-a-Service (IaaS) clouds have become more popular enabling users to run applications under virtual machines. Energy efficiency for IaaS clouds is still challenge. This paper investigates the energy-efficient scheduling problems of virtual machines (VMs) onto physical machines (PMs) in IaaS clouds along characteristics: multiple resources, fixed intervals and non-preemption of virtual machines. The scheduling problems are NP-hard. Most of existing works on VM placement reduce the total energy consumption by using the minimum number of active physical machines. There, however, are cases using the minimum number of physical machines results in longer the total busy time of the physical machines. For the scheduling problems, minimizing the total energy consumption of all physical machines is equivalent to minimizing total busy time of all physical machines. In this paper, we propose an scheduling algorithm, denoted as EMinTRE-LFT, for minimizing the total energy consumption of physical machines in the scheduling problems. Our extensive simulations using parallel workload models in Parallel Workload Archive show that the proposed algorithm has the least total energy consumption compared to the state-of-the-art algorithms.

**Keywords**—energy efficiency; energy-aware; power-aware; vm placement; IaaS; total busy time; fixed interval; fixed starting time; scheduling

## I. INTRODUCTION

Infrastructure-as-a-Service (IaaS) cloud [1] service provisions users with computing resources in terms of virtual machines (VMs) to run their applications [2], [3], [4]. These IaaS cloud systems are often built from virtualized data centers. Power consumption in a large-scale data centers requires multiple megawatts [5], [3]. Le et al. [3] estimate the energy cost of a single data center is more than \$15M per year. As these data centers has more physical servers, they will consume more energy. Therefore, advanced scheduling techniques for reducing energy consumption of these cloud systems are highly concerned for any cloud providers to reduce energy cost. Energy efficiency is an interesting research topic in cloud systems. Energy-aware scheduling of VMs in IaaS cloud is still challenging [2], [3], [6], [7].

Many previous works [8], [9] proved that the scheduling problems with fixed interval times are NP-hard. They [4],

[10] present techniques for consolidating virtual machines in cloud data centers by using bin-packing heuristics (such as First-Fit Decreasing [10], and/or Best-Fit Decreasing [4]). They attempt to minimize the number of running physical machines and to turn off as many idle physical machines as possible. Consider a  $d$ -dimensional resource allocation where each user requests a set of virtual machines (VMs). Each VM requires multiple resources (such as CPU, memory, and IO) and a fixed quantity of each resource at a certain time interval. Under this scenario, using a minimum of physical machines can result in increasing the total busy time of the active physical machines [11][9]. In a homogeneous environment where all physical servers are identical, the power consumption of each physical machine is linear to its CPU utilization [4], i.e., a schedule with longer working time will consume more energy than another schedule with shorter working time.

This paper presents a proposed heuristic, denoted as EMinTRE-LFT, to allocate VMs that request multiple resources in the fixed interval time and non-preemption into physical machines to minimize total energy consumption of physical machines while meeting all resource requirements. Using numerical simulations, we compare EMinTRE-LFT with the state-of-the-art algorithms include Power-Aware Best-Fit Decreasing (PABFD) [4], vector bin-packing norm-based greedy (VBP-Norm-L2) [10], and Modified First-Fit-Decreasing-Earliest (Tian-MFFDE) [9]. Using three parallel workload models [12], [13] and [14] in the Feitelson's Parallel Workloads Archive [15], the simulation results show that the proposed EMinTRE-LFT can reduce the total energy consumption of the physical servers by average of 23.7% compared with Tian-MFFDE [9]. In addition, EMinTRE-LFT can reduce the total energy consumption of the physical servers by average of 51.5% and respectively 51.2% compared with PABFD [4] and VBP-Norm-L2 [10]. Moreover, EMinTRE-LFT has also less total energy consumption than MinDFT-LDTF [11] in the simulation results.

The rest of this paper is structured as follows. Section II discusses related works. Section III describes the energy-

aware VM allocation problem with multiple requested resources, fixed starting and duration time. We also formulate the objective of scheduling, and present our theorems. The proposed EMinTRE-LFT algorithm presents in Section IV. Section V discusses our performance evaluation using simulations. Section VI concludes this paper and introduces future works.

## II. RELATED WORKS

The interval scheduling problems have been studied for many years with objective to minimizing total busy time. In 2007, Kovalyov et al. [16] has presented work to describe characteristics of a fixed interval scheduling problem in which each job has fixed starting time, fixed processing time, and is only processed in the fixed duration time on a available machine. The scheduling problem can be applied in other domains. Angelelli et al. [17] considered interval scheduling with a resource constraint in parallel identical machines. The authors proved the decision problem is NP-complete if number of constraint resources in each parallel machine is a fixed number greater than two. Flammini et al. [8] studied using new approach of minimizing total busy time to optical networks application. Tian et al. [9] proposed a Modified First-Fit Decreasing Earliest algorithm, denoted as Tian-MFFDE, for placement of VMs energy efficiency. The Tian-MFFDE sorts list of VMs in queue order by longest their running times first) and places a VM (in the sorted list) to any first available physical machine that has enough VM's requested resources. Our VM placement problem differs from these interval scheduling problems [16][17][9], where each VM requires for multiple resource (e.g. computing power, physical memory, network bandwidth, etc.) instead of all jobs in the interval scheduling problems are equally on demanded computing resource (i.e. each physical machine can process the maximum of  $g$  jobs in concurrently).

Energy-aware resource management in cloud virtualized data centers is critical. Many previous research [4], [18], [7], [19] proposed algorithms that consolidate VMs onto a small set of physical machines (PMs) in virtualized datacenters to minimize energy/power consumption of PMs. A group in Microsoft Research [10] has studied first-fit decreasing (FFD) based heuristics for vector bin-packing to minimize number of physical servers in the VM allocation problem. Some other works also proposed meta-heuristic algorithms to minimize the number of physical machines. Beloglazov's work [4] has presented a modified best-fit decreasing heuristic in bin-packing problem, denoted as PABFD, to place a new VM to a host. PABFD sorts all VMs in a decreasing order of CPU utilization and tends to allocate a VM to an active physical server that would take the minimum increase of power consumption. Knauth et al. [18] proposed the OptSched

scheduling algorithm to reduce cumulative machine up-time (CMU) by 60.1% and 16.7% in comparison to a round-robin and First-fit. The OptSched uses an minimum of active servers to process a given workload. In a heterogeneous physical machines, the OptSched maps a VM to a first available and the most powerful machine that has enough VM's requested resources. Otherwise, the VM is allocated to a new unused machine. In the VM allocation problem, however, minimizing the number of used physical machines is not equal to minimizing total of total energy consumption of all physical machines. Previous works do not consider multiple resources, fixed starting time and non-preemptive duration time of these VMs. Therefore, it is unsuitable for the power-aware VM allocation considered in this paper, i.g. these previous solutions can not result in a minimized total energy consumption for VM placement problem with certain interval time while still fulfilling the quality-of-service.

Chen et al [19] observed there exists VM resource utilization patterns. The authors presented an VM allocation algorithm to consolidate complementary VMs with spatial and temporal-awareness in physical machines. They introduce resource efficiency and use norm-based greedy algorithm, which is similar to in [10], to measure distance of each used resource's utilization and maximum capacity of the resource in a host. Their VM allocation algorithm selects a host that minimizes the value of this distance metric to allocate a new VM. Our proposed EMinTRE-LFT uses a different metric that unifies both increasing time and the  $L_2$ -norm of diagonal vector that is presenting available resources. In our proposed TRE metric, the increasing time is the difference between two total busy time of a PM after and before allocating a VM.

Our proposed EMinTRE-LFT algorithm that differs from these previous works. Our EMinTRE-LFT algorithm use the VM's fixed starting time and duration to minimize the total busy time on physical machines, and consequently minimize the total energy consumption in all physical servers. To the best of our knowledge, no existing works that surveyed in [20], [21], [22], [23] have thoroughly considered these aspects in addressing the problem of VM placement.

## III. PROBLEM DESCRIPTION

### A. Notations

We use the following notations in this paper:

$vm_i$ : The  $i^{th}$  virtual machine to be scheduled.

$M_j$ : The  $j^{th}$  physical machine.

$S$ : A feasible schedule.

$p_j^{min}$ : The minimum power consumed when  $M_j$  is 0% CPU utilization.

$p_j^{max}$ : The maximum power consumed when  $M_j$  is 100% CPU utilization.

$P_j(t)$ : Power consumption of  $M_j$  at a time point  $t$ .

$ts_i$ : Fixed starting time of  $vm_i$ .

$d_i$ : Duration time of  $vm_i$ .

$T$ : The maximum schedule length, which is the time that the last virtual machine will be finished.

$\mathcal{J}_j$ : Set of virtual machines that are allocated to  $M_j$  in the whole schedule.

$T_j^{busy}$ : The total busy time (ON time) of  $M_j$ .

$e_i$ : Energy consumption for running  $vm_i$  in the physical machine that  $vm_i$  is allocated.

$g$ : The maximum number of virtual machines that can be assigned to any physical machine.

### B. Power consumption model

Notations:

-  $U_j(t)$  is the CPU utilization of  $M_j$  at time  $t$ . -  $PE_j$  is the total number cores of  $M_j$ .

-  $mips_{i,c}$  is the allocated MIPS of the  $c^{th}$  processing element to the  $vm_i$  by  $M_j$ .

-  $MIPS_{j,c}$  is the maximum computing power (in MIPS) of the  $c^{th}$  core on  $M_j$ .

In this paper, we use the following energy consumption model proposed in [5][4] for a physical machine. Let call  $\alpha = P_j^{min}/P_j^{max}$  is fraction of the minimum power consumed when  $M_j$  is idle (0% CPU utilization) and the maximum power consumed when the physical machine is fully utilized (100% CPU utilization). The power consumption of  $M_j$ , denoted as  $P_j(\cdot)$  with  $(j = 1, 2, \dots, m)$ , is formulated as follow:

$$P_j(t) = (\alpha + (1 - \alpha) \cdot U_j(t)) \cdot P_j^{max} \quad (1)$$

We assume that all cores in CPU are homogeneous, i.e.  $\forall c = 1, 2, \dots, PE_j : MIPS_{j,c} = MIPS_{j,1}$ . The CPU utilization  $U_j(t)$  is formulated as follow:

$$U_j(t) = \left( \frac{1}{PE_j \times MIPS_{j,1}} \right) \sum_{c=1}^{PE_j} \sum_{vm_i \in \mathcal{J}_j} mips_{i,c} \quad (2)$$

The energy consumption of the  $M_j$  in the time period  $[t_1, t_2]$  denoted as  $\Delta E_j$  with CPU utilization  $U_j$  is formulated as follow:

$$\Delta E_j = P_j(U_j) \cdot (t_2 - t_1) = (\alpha \cdot P_j^{max} + (1 - \alpha) \cdot P_j^{max} \cdot U_j) \cdot \Delta T \quad (3)$$

where:

$\Delta T_j$ : The busy time of  $M_j$  that is defined as:  $\Delta T_j = (t_2 - t_1)$ .

Assume that a virtual machine  $vm_i$  changes the CPU utilization is  $\Delta u_j$  for during  $[t_1, t_2]$  and the  $vm_i$  uses full utilization of its requested resources in the worst case on  $M_j$ . The energy consumption by the  $vm_i$ , denoted as  $e_i$ , is formulated as:

$$e_i = (1 - \alpha) \cdot P_j^{max} \cdot \Delta u_j \cdot (t_2 - t_1) \quad (4)$$

Let  $T_j^{busy}$  be the total busy time of  $M_j$ , let  $e_i$  be energy consumed by  $vm_i$ , and let  $vm_i \in M_j$  be set of virtual machines  $vm_i$  ( $i = 1, 2, \dots, n$ ) that are allocated to  $M_j$  in the whole schedule. Let  $E_j$  be the total energy consumed by  $M_j$  and  $E_j$  is the sum of energy consumption  $\Delta E_j$  during the total busy time  $T_j^{busy}$  that is formulated as:

$$E_j = (\alpha \cdot P_j^{max} + (1 - \alpha) \cdot P_j^{max} \cdot U_j) \cdot T_j^{busy} \quad (5)$$

where  $\alpha \cdot P_j^{max} \cdot T_j^{busy}$  is called the base (ON) energy consumption for  $M_j$  during the total busy time, i.e.,  $E_j^{base} = \alpha \cdot P_j^{max} \cdot T_j^{busy}$ , and  $((1 - \alpha) \cdot P_j^{max} \cdot U_j \cdot T_j^{busy})$  is the increasing energy consumed by some VMs scheduled to  $M_j$ .

$$E_j = \alpha \cdot P_j^{max} \times T_j^{busy} + \sum_{vm_i \in M_j} e_i \quad (6)$$

### C. Problem formulation

Consider the following scheduling problem. We are given a set of  $n$  virtual machines  $\mathcal{J} = \{vm_1, \dots, vm_n\}$  to be scheduled on a set of  $m$  identical physical servers  $\mathcal{M} = \{M_1, \dots, M_m\}$ , each server can host a maximum number of  $g$  virtual machines. Each VM needs  $d$ -dimensional demand resources in a fixed interval with non-migration. Each  $vm_i$  is started at a fixed starting time ( $ts_i$ ) and is non-preemptive during its duration time ( $d_i$ ). Types of resource considered in the problem include computing power (i.e., the total Million Instruction Per Seconds (MIPS) of all cores in a physical machine), physical memory (i.e., the total MBytes of RAM in a physical machine), network bandwidth (i.e., the total Kb/s of network bandwidth in a physical machine), and storage (i.e., the total free GBytes of file system in a physical machine), etc.

The objective is to find out a feasible schedule  $S$  that minimizes the total energy consumption in the equation (8) with  $\forall i \in \{1, 2, \dots, n\}, \forall j \in \{1, 2, \dots, m\}, \forall t \in [0, T]$  as following:

$$\text{Min} \left( \sum_{j=1}^m (\alpha \times P_j^{max} \times T_j^{busy}) + \sum_{i=1}^n e_i \right) \quad (7)$$

where:

-  $\alpha = P_j^{min}/P_j^{max}$  is the fraction of idle power and maximum power consumption by physical machine  $M_j$ .

-  $T_j^{busy}$  is the total busy time of  $M_j$ .

In homogeneous physical machines (PMs), all PMs have the same idle power and maximum power consumption. Therefore  $\alpha$  is the same for all PMs. We rewrite the objective scheduling as following:

$$\text{Min} \left( \alpha \times P^{max} \times \sum_{j=1}^m T_j^{busy} + \sum_{i=1}^n e_i \right) \quad (8)$$

The scheduling problem has the following hard constraints that are described in our previous work [11] as following:

- Constraint 1: Each VM is only processed by a physical server at any time with non-migration and non-preemption.
- Constraint 2: Each VM does not request any resource larger than the maximum total capacity resource of any physical server.
- Constraint 3: The sum of total demand resources of these allocated VMs is less than or equal to the total capacity of the resources of  $M_j$ .

#### D. Preliminaries

**Definition 1 (Length of intervals.):** Given a time interval  $I = [s, f]$ , the length of  $I$  is  $len(I) = f - s$ . Extensively, to a set  $\mathcal{J}$  of intervals, length of  $\mathcal{J}$  is  $len(\mathcal{J}) = \sum_{I \in \mathcal{J}} len(I)$ .

**Definition 2 (Span of intervals.):** For a set  $\mathcal{J}$  of intervals, we define the span of  $\mathcal{J}$  as  $span(\mathcal{J}) = len(\cup \mathcal{J})$ .

**Definition 3 (Optimal schedule):** An optimal schedule is the schedule that minimizes the total busy time of physical machines. For any instance  $\mathcal{J}$  and parameter  $g \geq 1$ ,  $OPT(\mathcal{J}, g)$  denotes the cost of an optimal schedule.

In this paper, we denote  $\mathcal{J}$  is set of time intervals that derived from given set of all requested VMs. In general, we use instance  $\mathcal{J}$  is alternative meaning to a given set of all requested VMs in context of this paper.

**Observations: Cost, capacity, span bounds.** For any instance  $\mathcal{J}$ , which is set of time intervals derived from given set of all requested VMs, and capacity parameter  $g \geq 1$ , which is the maximum number of VMs that can be allocated on any physical machine, the following bounds are held:

- The optimal cost bound:  $OPT(\mathcal{J}, g) \leq len(\mathcal{J})$ .
- The capacity bound:  $OPT(\mathcal{J}, g) \geq \frac{len(\mathcal{J})}{g}$ .
- The span bound:  $OPT(\mathcal{J}, g) \geq span(\mathcal{J})$ .

For any feasible schedule  $s$  on a given set of virtual machines, the total busy time of all physical machines that are used in the schedule  $s$  is bounded by the maximum total length of all time intervals in a given instance  $\mathcal{J}$ . Therefore, the optimal cost bound holds because  $OPT(\mathcal{J}, g) = len(\mathcal{J})$  iff all intervals are non-overlapping, i.e.,  $\forall I_1, I_2 \in \mathcal{J}$  then  $I_1 \cap I_2 = \emptyset$ .

Intuitively, the capacity bound holds because  $OPT(\mathcal{J}, g) = \frac{len(\mathcal{J})}{g}$  iff, for each physical server, exactly  $g$  VMs are neatly scheduled in that physical server. The span bound holds because at any time  $t \in \cup \mathcal{J}$  at least one machine is working.

#### E. Theorems

In the following theorems, all physical machines are homogeneous. Let  $P^{min}$  and  $P^{max}$  are the minimum/idle

power and maximum power consumption of a physical machine respectively. We have  $\alpha = P^{min}/P^{max}$ .

**Theorem 1:** Minimizing total energy consumption in (8) is equivalent to minimizing the sum of total busy time of all physical machines ( $\sum_{j=1}^m T_j^{busy}$ ).

$$\text{Min } (\alpha \times P^{max} \times \sum_{j=1}^m T_j^{busy} + \sum_{i=1}^n e_i) \sim \text{Min } (\sum_{j=1}^m T_j^{busy}) \quad (9)$$

*Proof:* A proof for this theorem see detail in [11]. ■

Based on the above theorem, we propose our energy-aware algorithms denoted as EMinTRE-LFT which is presented in the next section.

**Definition 4:** For any schedule we denote by  $\mathcal{J}_j$  the set of virtual machines allocated to the physical machine  $M_j$  by the schedule. Let  $T_j$  denote the total busy time of  $M_j$  is the span of  $\mathcal{J}_j$ , i.e.,  $T_j = span(\mathcal{J}_j)$ .

**Definition 5:** For any instance  $\mathcal{J}$ , the total busy time of the entire schedule of  $\mathcal{J}$  computed by the algorithm  $H$ , denoted as  $cost^H(\mathcal{J})$ , is defined as  $cost^H(\mathcal{J}) = \int_0^{span(\mathcal{J})} N^H(t) dt$ , where as  $N^H(t)$  is the number of physical machines used at the time  $t$  by the algorithm  $H$ .

**Definition 6:** For any instance  $\mathcal{J}$  and parameter  $g \geq 1$ ,  $E^{OPT}(\mathcal{J}, g)$ , which is denoted as the minimized total energy consumption of all physical machines in an optimal schedule for the  $\mathcal{J}$ , is formulated as:  $E^{OPT}(\mathcal{J}, g) = \alpha \times P^{max} \cdot OPT(\mathcal{J}, g) + \sum_{i=1}^n e_i$ .

**Theorem 2:** For any instance  $\mathcal{J}$ , the lower and upper of the total energy consumption in an optimal schedule are bounded by:  $P_{min} \cdot \frac{len(\mathcal{J})}{g} \leq E^{OPT}(\mathcal{J}, g) \leq P^{max} \cdot len(\mathcal{J})$ .

*Proof:* For any instance  $\mathcal{J}$ , let  $OPT(\mathcal{J}, g)$  be the total busy time of the optimal schedule for the  $\mathcal{J}$ , and let  $E^*$  be the total energy consumption for the optimal schedule for the  $\mathcal{J}$ .

The total energy consumption of an optimal schedule needs to account for all physical machines running during  $OPT(\mathcal{J}, g)$ . We have:  $E^* = P_{min} \cdot OPT(\mathcal{J}, g) + \sum_{i=1}^n e_i$ .

From Definition 6, we have  $E^{OPT}(\mathcal{J}, g) = E^*$ .

Apply the capacity bound in Theorem III-D, we have  $OPT(\mathcal{J}, g) \geq \frac{len(\mathcal{J})}{g}$ . Thus,  $E^* \geq P_{min} \cdot \frac{len(\mathcal{J})}{g} + \sum_{i=1}^n e_i$ .

Recall that the energy consumption of each virtual machine is non-negative, thus  $e_i > 0$ . Therefore,  $E^* \geq P_{min} \cdot \frac{len(\mathcal{J})}{g}$ . Thus

$$E^{OPT}(\mathcal{J}, g) \geq P_{min} \cdot \frac{len(\mathcal{J})}{g} \quad (10)$$

We prove the upper bound of the minimized total energy consumption as following. Apply the optimal cost bound in Theorem III-D, we have  $OPT(\mathcal{J}, g) \leq len(\mathcal{J})$ .

Thus

$$E^* \leq P_{min} \cdot len(\mathcal{J}) + \sum_{i=1}^n e_i. \quad (11)$$



Apply the linear power consumption as in the Equation (1) and Equation (3), the energy consumption of each  $i$ -th virtual machine in period time of  $[ts_i, ts_i + d_i]$  that denotes as  $e_i$  is:

$$e_i = \int_{ts_i}^{ts_i + d_i} P_j(U_{vm_i}) dt = (P_j^{max} - P_j^{idle}) \cdot U_{vm_i} \cdot d_i$$

where  $U_{vm_i}$  is the percentage of CPU usage of the  $i$ -th virtual machine on a  $j$ -th physical machine.

Because any virtual machine always requests CPU usage lesser than or equal to the maximum total capacity CPU of every physical machine, i.e.,  $U_{vm_i} \leq 1$ .

$$\Rightarrow e_i \leq (P_j^{max} - P_j^{idle}) \cdot d_i$$

Note that in this proof, all physical machines are identical with same power consumption model thus  $P^{max}$  and  $P^{idle}$  are the maximum power consumption and the idle power consumption of each physical machine. Thus:

$$e_i \leq (P^{max} - P^{idle}) \cdot d_i$$

Let  $I_i$  is interval of each  $i$ -th virtual machine,  $I_i = [ts_i, ts_i + d_i]$ . By the definition the length of interval is  $len(I_i) = d_i$  that is duration time of each  $i$ -th virtual machine. Thus:

$$e_i \leq (P^{max} - P^{idle}) \cdot len(I_i)$$

The total energy consumption of  $n$  virtual machines is formulated as:

$$\begin{aligned} \sum_{i=1}^n e_i &\leq \sum_{i=1}^n [(P^{max} - P^{idle}) \cdot len(I_i)] \\ \Leftrightarrow \sum_{i=1}^n e_i &\leq (P^{max} - P^{idle}) \cdot \sum_{i=1}^n len(I_i) \\ \Leftrightarrow \sum_{i=1}^n e_i &\leq (P^{max} - P^{idle}) \cdot len(\mathcal{J}). \end{aligned} \quad (12)$$

From Equation (11), we have:

$$\begin{aligned} E^* &\leq P_{min} \cdot len(\mathcal{J}) + \sum_{i=1}^n e_i \\ E^* &\leq P_{min} \cdot len(\mathcal{J}) + (P^{max} - P^{idle}) \cdot len(\mathcal{J}) \end{aligned}$$

$$E^* \leq (P_{min} + (P^{max} - P^{idle})) \cdot len(\mathcal{J}) \quad (13)$$

By the definition, the unit energy of a physical machine equals to the idle power consumption in the unit time, i.e.,  $P_{min} = P^{idle}$ . From the Equation (13):

$$E^* \leq P^{max} \cdot len(\mathcal{J}) \quad (14)$$

$$\Leftrightarrow E^{OPT}(\mathcal{J}, g) \leq P^{max} \cdot len(\mathcal{J}) \quad (15)$$

From both of two equations (10) and (15), we have:

$$P_{min} \cdot \frac{len(\mathcal{J})}{g} \leq E^{OPT}(\mathcal{J}, g) \leq P^{max} \cdot len(\mathcal{J}) \quad (16)$$

We prove the theorem.  $\blacksquare$

## IV. SCHEDULING ALGORITHMS

### A. EMinTRE-LFT scheduling algorithm

In this section, we present the proposed energy-aware scheduling algorithm, denoted as EMinTRE-LFT, with pseudo-code of EMinTRE-LFT in Algorithm 1. Algorithm EMinTRE-LFT has two (2) steps: sorts the list of virtual machines in order decreasing finishing time first. Next, EMinTRE-LFT allocates the first next virtual machine  $i$  to the first physical machine  $M_j$  such that  $M_j$  has enough resource to provision the virtual machine  $i$  and TRE metric of  $M_j$  denoted as  $TRE_j$  is minimum. The  $TRE_j$  is formulated as in the following equation 19. The EMinTRE-LFT solves these scheduling problems in time complexity of  $\mathcal{O}(n \times m \times q)$  where  $n$  is the number of VMs to be scheduled,  $m$  is the number of physical machines, and  $q$  is the maximum number of allocated VMs in the physical machines  $M_j, \forall j = 1, 2, \dots, m$ .

Based on the equation 2, the utilization of a resource  $r$  (resource  $r$  can be cores, computing power, physical memory, network bandwidth, storage, etc.) of the  $M_j$ , denoted as  $U_{j,r}$ , is formulated as:

$$U_{j,r} = \sum_{s \in n_j} \frac{V_{s,r}}{H_{j,r}}. \quad (17)$$

where  $n_j$  is the list of virtual machines that are assigned to the  $M_j$ ,  $V_{s,r}$  is the amount of requested resource  $r$  of the virtual machine  $s$  (note that in our study the value of  $V_{s,r}$  is fixed for each user request), and  $H_{j,r}$  is the maximum capacity of the resource  $r$  in  $M_j$ .

The available resource is presented using diagonal vector, where the  $L2$ -norm of the diagonal vector (denoted as  $D_j$ ) is formulated as:

$$D_j = \sqrt{\left( \sum_{r \in \mathcal{R}} ((1 - U_{j,r}) \times w_r)^2 \right)} \quad (18)$$

where  $\mathcal{R}$  is the set of resource types in a host ( $\mathcal{R} = \{\text{core, mips, ram, netbw, io, storage}\}$ ) and  $w_r$  is weight of resource  $r$  in a physical machine.

In this paper, we propose the TRE metric for the increasing total busy time and the  $L2$ -norm of the diagonal vector ( $D_j$ ) of the physical machine  $j$ -th that is calculated as:

$$TRE_j = \left( \frac{t^{diff} \times w_{r=time}}{T_j^{busy}} \right)^2 + D_j^2 \quad (19)$$

## V. PERFORMANCE EVALUATION

### A. Algorithms

In this section, we study the following VM allocation algorithms:

- PABFD, a power-aware and modified best-fit decreasing heuristic [4]. The PABFD sorts the list of  $VM_i$  ( $i=1, 2, \dots, n$ ) by their total requested CPU utilization,

---

**Algorithm 1 : EMinTRE-LFT: Energy-aware Greedy-based Scheduling Algorithm**


---

```

1: function EMINTRE-LFT
2:   Input: vmList - a list of virtual machines to be
      scheduled, hostList - a list of physical servers
3:   Output: a feasible schedule or null
4:   vmList = sortVmListByOrderLastestFinishingTime-
      First( vmList )                                ▷
      1
5:   m = hostList.size(); n = vmList.size();
6:   T[j] = 0,  $\forall j \in [1, m]$ 
7:   for i = 1 to n do                                ▷ on the VMs list
8:     vm = vmList.get(i)
9:     allocatedHost = null
10:    T1 = sumTotalHostBusyTime( T )
11:    minRETime =  $+\infty$ 
12:    for j = 1 to m do                                ▷ on the hosts list
13:      host = hostList.get( j )
14:      hostVmList = sortVmListByOrder(
        host.getVms(), order=[finishtime])
15:      if host.checkAvailableResource( vm ) then
16:
17:        preTime = T[ host.id ]
18:        T[ host.id ] =
        host.estimateHostTotalCompletionTime( vm )
19:        T2 = sumTotalHostBusyTime( T )
20:        diffTime = Math.max( T2 - T1, 0 )
21:        TRE = EstimateMetricTimeResEff( diff-
        Time, host )
22:        if (minTRE > TRE) then
23:          minRETime = TRE
24:          allocatedHost = host
25:        end if
26:        T[ host.id ] = preTime    ▷ Next iterate
        over the hostList and choose the host that minimize
        the value of different time and resource efficiency
27:      end if
28:    end for
29:    if (allocatedHost != null) then
30:      allocate the vm to the host
31:      add the pair of vm (key) and host to the
        mapping
32:    else
33:      "Cannot allocate the virtual machine vm."
34:    end if
35:  end for
36:  return mapping
37: end function
38: sumTotalHostBusyTime(T[]) =  $\sum_{j=1}^m T_j$     ▷ T[1...m]:
      Array of total completion times of m physical servers

```

---

and assigns new VM to any host that has a minimum

---

**Algorithm 2 Estimating the metric for increasing time and resource efficiency**


---

```

1: function ESTIMATEMETRICTIMESEFF
2:   Input: (tdiff, host) - tdiff is a different time, host
      is a candidate physical machine
3:   Output: TRE - a value of metric time and resource
      efficiency
4:   Set  $\mathcal{R} = \{\text{cores, mips, ram, io, netbw, storage}\}$ 
5:   j = host.getId(); nj = host.getVmList();
6:   for r ∈  $\mathcal{R}$  do
7:     Calculate the resource utilization, Uj,r as in the
      Equation (17).
8:   end for
9:   weights[] ← Read weight of resources from con-
      figuration file
10:  Calculate the TREj metric of host j as in the
      equation (19)
11:   $D_j = \sqrt{\sum_{r \in \mathcal{R}} ((1 - U_{j,r}) \times w_r)^2}$ 
12:   $TRE_j = (\frac{t^{diff} \times w_{time}}{T_j^{busy}})^2 + D_j^2$  ▷ wtime is weight of
      the different time
13:  return TREj
14: end function

```

---

increase in power consumption.

- VBP-Norm-L2, a vector packing heuristics that is presented as Norm-based Greedy with degree 2 [10]. Weights of these Norm-based Greedy heuristics use FFDavgSum which are  $exp(x)$ , which is the value of the exponential function at the point *x*, where *x* is average of sum of demand resources (e.g. CPU, memory, storage, network bandwidth, etc.). VBP-Norm-L2 assigns new VM to any host that has minimum of these norm values.
- MinDFT-LDTF: the algorithm sorts list of *VM<sub>i</sub>* (*i*=1, 2,..., *n*) by their starting time (*ts<sub>i</sub>*) and respectively by their finished time (*ts<sub>i</sub>* + *dur<sub>i</sub>*), then MinDFT-LDTF allocates each VM (in a given sorted list of VMs) to a host that has a minimum increase in total completion times of hosts as in algorithm MinDFT [11].
- EMinTRE-LDTE, the algorithm is proposed in the Section IV.

### B. Methodology

We evaluate these algorithms by simulation using the CloudSim [24] to create simulated cloud data center systems that have identical physical machines, heterogeneous VMs, and with thousands of CloudSim's cloudlets [24] (we assume that each HPC job's task is modeled as a cloudlet that is run on a single VM). The information of VMs (and also cloudlets) in these simulated workloads is

Table I  
EIGHT (08) VM TYPES IN SIMULATIONS

VM Type	MIPS	Cores	Memory (Unit: MBytes)	Network (Unit: Mbits/s)	Storage (Unit: GBytes)
Type 1	2500	8	6800	100	1000
Type 2	2500	2	1700	100	422.5
Type 3	3250	8	68400	100	1000
Type 4	3250	4	34200	100	845
Type 5	3250	2	17100	100	422.5
Type 6	2000	4	15000	100	1690
Type 7	2000	2	7500	100	845
Type 8	1000	1	1875	100	211.25

Table II  
INFORMATION OF A TYPICAL PHYSICAL MACHINE (HOST) WITH 16 CORES CPU (3250 MIPS/CORE), 136.8 GBYTES OF AVAILABLE PHYSICAL MEMORY, 10 GB/s OF NETWORK BANDWIDTH, 10 TBYTES OF STORAGE AND IDLE, MAXIMUM POWER CONSUMPTION IS 175, 250 (W).

Type	MIPS	Cores	Memory (Unit: MBytes)	Network (Unit: Mbits/s)	Storage (Unit: GBytes)	$p_{idle}$ (Unit: Watts)	$p_{max}$ (Unit: Watts)
M1	3250	16	140084	10000	10000	175	250

Table III  
THE NORMALIZED TOTAL ENERGY CONSUMPTION. SIMULATION RESULTS OF SCHEDULING ALGORITHMS SOLVING SCHEDULING PROBLEMS WITH 12681 VMs AND 5000 PMs USING FEITELSON'S PARALLEL WORKLOAD MODEL [12]. ALGORITHM EMINTRE-LFT wtX HAS WEIGHT OF TIME IS EQUAL TO X ( $X = 1; 0.1; 0.001$ ).

Algorithm	Energy (Unit: kWh)	Norm. Energy	Saving Energy (+;better;-worst)
PABFD	1,055.42	1.598	-60%
VBP-Norm-L2	1,054.69	1.597	-60%
MinDFT-LDTF	603.90	0.915	9%
Tian-MFFDE	660.30	1.000	0%
EMinTRE-LFT wt1	503.43	0.762	24%
EMinTRE-LFT wt0.01	503.43	0.762	24%
EMinTRE-LFT wt0.001	503.43	0.762	24%

Table IV  
THE NORMALIZED TOTAL ENERGY CONSUMPTION. SIMULATION RESULTS OF SCHEDULING ALGORITHMS SOLVING SCHEDULING PROBLEMS WITH 15,201 VMs AND 5,000 PMs USING DOWNEY97'S PARALLEL WORKLOAD MODEL [13] IN THE PARALLEL WORKLOAD ARCHIVE [15]. ALGORITHM EMINTRE-LFT wtX HAS WEIGHT OF TIME IS EQUAL TO X ( $X = 1; 0.1; 0.001$ ).

Algorithm	Energy (Unit: kWh)	Norm. Energy	Saving Energy (+;better;-worst)
PABFD	878.01	1.523	-52.3%
Norm-VBP-L2	876.49	1.520	-52.0%
Tian-MFFDE	576.55	1.000	0.0%
MinDFT-LDTF	502.61	0.872	12.8%
EMinTRE-LFT wt1	416.35	0.722	27.8%
EMinTRE-LFT wt0.01	416.35	0.722	27.8%
EMinTRE-LFT wt0.001	416.35	0.722	27.8%

Table V  
THE NORMALIZED TOTAL ENERGY CONSUMPTION. SIMULATION RESULTS OF SCHEDULING ALGORITHMS SOLVING SCHEDULING PROBLEMS WITH 8847 VMs AND 5000 PHYSICAL MACHINES (HOSTS) USING LUBLIN99'S PARALLEL WORKLOAD MODEL [14]

Algorithm	Energy (Unit: kWh)	Norm. Energy	Saving Energy (+;better;-worst)
PABFD	460.66	1.601	-60.1%
Norm-VBP-L2	453.23	1.575	-57.5%
Tian-MFFDE	287.78	1.000	0.0%
MinDFT-LDTF	263.86	0.917	8.3%
EMinTRE-LFT wt0.001	232.29	0.807	19.3%
EMinTRE-LFT wt0.01	232.29	0.807	19.3%
EMinTRE-LFT wt1	232.29	0.807	19.3%

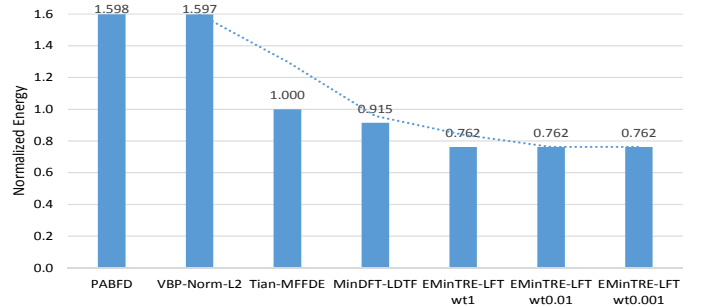


Figure 1. The normalized total energy consumptions compare to Tian-MFFDE. Simulation result for scheduling algorithms with Feitelson's parallel workload model [12] in the Parallel Workload Archive [15] that includes 1,000 jobs have total of 12,681 VMs and 5000 PMs.

extracted from two parallel job models are Feitelson's parallel workload model [12], Downey98's parallel workload model [13] and Lublin99's parallel workload model [14] in Parallel Workloads Archive (PWA) [15]. When converting from the generated log-trace files, each cloudlet's length is a product of the system's processing time and CPU rating (we set the CPU rating is equal to included VM's MIPS). We convert job's submission time, job's start time

(if the start time is missing, then the start time is equal to sum of job's submission time and job's waiting time), job's request run-time, and job's number of processors in job data from the log-trace in PWA [15] to VM's submission time, starting time and duration time, and number of VMs (each VM is created in round-robin in the four types of VMs in Table I on the number of VMs). Eight (08) types of VMs as presented in the

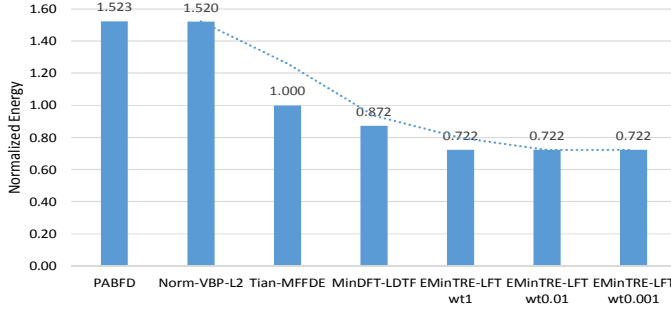


Figure 2. The normalized total energy consumptions compare to Tian-MFFDE. Simulation result for scheduling algorithms with Downey97's parallel workload model [13] in the Parallel Workload Archive [15] that includes 1,000 jobs have total of 15,201 VMs and 5000 PMs.

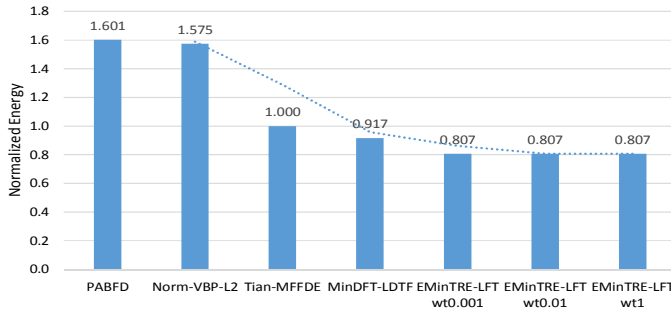


Figure 3. The normalized total energy consumption compare to Tian-MFFDE. Result of simulations with Lublin99's parallel workload model [14] that includes 1,000 jobs have total of 8,847 VMs and 5,000 PMs.

Table I are used in the [9] that are similar to categories in Amazon EC2's VM instances: high-CPU VM, high-memory VM, small VM, and micro VM, etc.. All physical machines are identical and each physical machine is a typical physical machine (Hosts) with 16 cores CPU (3250 MIPS/core), 136.8 GBytes of available physical memory, 10 Gb/s of network bandwidth, 10 TBytes of available storage. The minimum and maximum power consumed of each physical machine is 175W and 250W respectively (the minimum power when a PM idle is  $175:250 = 70\%$  of the maximum power consumption as in [5][4]). In the simulations, we use weights as following: (i) weight of increasing time for mapping a VM to PM: {0.001, 0.01, 1}; (ii) weights of computing resources such as number of MIPS per CPU core, physical memory (RAM), network bandwidth, and storage respectively are equally to 1. We denoted EMinTRE-LFT wt0.001, EMinTRE-LFT wt0.01 and EMinTRE-LFT wt1 as the total energy consumption of algorithm EMinTRE-LFT in the simulations has weight of increasing time for mapping a VM to PM is {0.001, 0.01, 1} respectively.

We choose Modified First-Fit Decreasing Earliest (denoted as Tian-MFFDE) [9] as the baseline because Tian-MFFDE is the best algorithm in the energy-aware scheduling algorithm to time interval scheduling. We also com-

pare our proposed VM allocation algorithms with PABFD [4] because the PABFD is a famous power-aware best-fit decreasing in the energy-aware scheduling research community, and a vector bin-packing algorithm (VBP-Norm-L2) to show the importance of with/without considering VM's starting time and finish time in reducing the total energy consumption of VM placement problem.

### C. Results and Discussions

The simulation results are shown in the three tables (Table III, Table IV and Table V) and figures. Three (03) figures include Fig. 1, Fig. 2 and Fig. 3 show bar charts comparing energy consumption of VM allocation algorithms that are normalized with the Tian-MFFDE. None of the scheduling algorithms use VM migration techniques, and all of them satisfy the Quality of Service (e.g. the scheduling algorithm provisions maximum of user VM's requested resources). We use total energy consumption as the performance metric for evaluating these VM allocation algorithms.

Using three parallel workload models [12], [13] and [14] in the Feitelson's Parallel Workloads Archive [15], the simulation results show that the proposed EMinTRE-LFT can reduce the total energy consumption of the physical servers by average of 23.7% compared with Tian-MFFDE [9]. In addition, EMinTRE-LFT can reduce the total energy consumption of the physical servers by average of 51.5% and respectively 51.2% compared with PABFD [4] and VBP-Norm-L2 [10]. Moreover, EMinTRE-LFT has also less total energy consumption than MinDFT-LDTF [11] in the simulation results.

## VI. CONCLUSIONS AND FUTURE WORK

In this paper, we formulated an energy-aware VM allocation problem with multiple resource, fixed interval and non-preemption constraints. We also discussed our key observation in the VM allocation problem, i.e., minimizing total energy consumption is equivalent to minimize the sum of total completion time of all physical machines (PMs). Our proposed algorithm EMinTRE-LFT can all reduce the total energy consumption of the physical servers compared with the state-of-the-art algorithms in simulation results on three parallel workload models of Feitelson's [12], Downey98's [13], and Lublin99's [14].

We are developing the algorithm EMinTRE-LFT into a cloud resource management software (e.g. OpenStack Nova Scheduler). In the future, we would like to evaluate more with the weights of increasing time and  $L2$ -norm of diagonal vector on available resources. Additionally, we are working on IaaS cloud systems with heterogeneous physical servers and job requests consisting of multiple VMs using EPOBF [6]. We are studying how to choose the right weights of time and resources (e.g. computing



power, physical memory, network bandwidth, etc.) in Machine Learning techniques.

#### ACKNOWLEDGMENT

#### REFERENCES

- [1] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: state-of-the-art and research challenges," *Journal of Internet Services and Applications*, vol. 1, no. 1, pp. 7–18, apr 2010.
- [2] S. K. Garg, C. S. Yeo, A. Anandasivam, and R. Buyya, "Energy-Efficient Scheduling of HPC Applications in Cloud Computing Environments," *CoRR*, vol. abs/0909.1146, 2009.
- [3] K. Le, R. Bianchini, J. Zhang, Y. Jaluria, J. Meng, and T. D. Nguyen, "Reducing electricity cost through virtual machine placement in high performance computing clouds," in *SC*, 2011, p. 22.
- [4] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," *Future Generation Comp. Syst.*, vol. 28, no. 5, pp. 755–768, 2012.
- [5] X. Fan, W.-D. Weber, and L. Barroso, "Power provisioning for a warehouse-sized computer," in *ISCA*, 2007, pp. 13–23.
- [6] N. Quang-Hung, N. Thoai, and N. T. Son, "EPOBF: Energy Efficient Allocation of Virtual Machines in High Performance Computing Cloud," *TLDKS XVI*, vol. LNCS 8960, pp. 71–86, 2014.
- [7] I. Takouna, W. Dawoud, and C. Meinel, "Energy Efficient Scheduling of HPC-jobs on Virtualize Clusters using Host and VM Dynamic Configuration," *Operating Systems Review*, vol. 46, no. 2, pp. 19–27, 2012.
- [8] M. Flammini, G. Monaco, L. Moscardelli, H. Shachnai, M. Shalom, T. Tamir, and S. Zaks, "Minimizing total busy time in parallel scheduling with application to optical networks," *Theoretical Computer Science*, vol. 411, no. 40-42, pp. 3553–3562, Sep. 2010.
- [9] W. Tian and C. S. Yeo, "Minimizing total busy time in offline parallel scheduling with application to energy efficiency in cloud computing," *Concurrency and Computation: Practice and Experience*, vol. 27, no. 9, pp. 2470–2488, jun 2013.
- [10] R. Panigrahy, K. Talwar, L. Uyeda, and U. Wieder, "Heuristics for Vector Bin Packing," Microsoft Research, Tech. Rep., 2011.
- [11] N. Quang-Hung, D.-K. Le, N. Thoai, and N. T. Son, "Heuristics for Energy-Aware VM Allocation in HPC Clouds," *Future Data and Security Engineering (FDSE 2014)*, vol. 8860, pp. 248–261, 2014.
- [12] D. G. Feitelson, "Packing schemes for gang scheduling," in *Job Scheduling Strategies for Parallel Processing*. Springer, 1996, pp. 89–110.
- [13] A. B. Downey, "A parallel workload model and its implications for processor allocation," *Cluster Computing*, vol. 1, no. 1, pp. 133–145, 1998.
- [14] U. Lublin and D. G. Feitelson, "The workload on parallel supercomputers: modeling the characteristics of rigid jobs," *Journal of Parallel and Distributed Computing*, vol. 63, no. 11, pp. 1105–1122, 2003.
- [15] D. G. Feitelson, "Parallel Workloads Archive," (retrieved on 31 Januray 2014), <http://www.cs.huji.ac.il/labs/parallel/workload/>.
- [16] M. Y. Kovalyov, C. Ng, and T. E. Cheng, "Fixed interval scheduling: Models, applications, computational complexity and algorithms," *European Journal of Operational Research*, vol. 178, no. 2, pp. 331–342, 2007.
- [17] E. Angelelli and C. Filippi, "On the complexity of interval scheduling with a resource constraint," *Theoretical Computer Science*, vol. 412, no. 29, pp. 3650–3657, 2011.
- [18] T. Knauth and C. Fetzer, "Energy-aware scheduling for infrastructure clouds," in *4th IEEE International Conference on Cloud Computing Technology and Science Proceedings*. IEEE, Dec. 2012, pp. 58–65.
- [19] L. Chen and H. Shen, "Consolidating complementary VMs with spatial/temporal-awareness in cloud datacenters," in *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*. IEEE, Apr. 2014, pp. 1033–1041.
- [20] A. Beloglazov, R. Buyya, Y. C. Lee, and A. Zomaya, "A Taxonomy and Survey of Energy-Efficient Data Centers and Cloud Computing Systems," *Advances in Computers*, vol. 82, pp. 1–51, 2011.
- [21] A.-C. Orgerie, M. D. de Assuncao, and L. Lefevre, "A survey on techniques for improving the energy efficiency of large-scale distributed systems," *ACM Computing Surveys*, vol. 46, no. 4, pp. 1–31, Mar. 2014.
- [22] A. Hameed, A. Khoshkbarforoushha, R. Ranjan, P. P. Jayaraman, J. Kolodziej, P. Balaji, S. Zeadally, Q. M. Malluhi, N. Tziritas, A. Vishnu, S. U. Khan, and A. Zomaya, "A survey and taxonomy on energy efficient resource allocation techniques for cloud computing systems," pp. 1–24, Jun. 2014.
- [23] T. Mastelic, A. Oleksiak, H. Claussen, I. Brandic, J.-M. Pierson, and A. V. Vasilakos, "Cloud computing: Survey on energy efficiency," *ACM Comput. Surv.*, vol. 47, no. 2, pp. 33:1–33:36, Dec. 2014.
- [24] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Softw., Pract. Exper.*, vol. 41, no. 1, pp. 23–50, 2011.